AMATH 515 HOMEWORK 4: COMPUTATIONAL REPORT

WINNIE LAU

AMATH Department, University of Washington, Seattle, WA
wwlau@uw.edu

1. Introduction

This report compares the algorithms steepest descent, Newton's Method and Quasi-Newton methods, specifically DFP and BFGS, using bisection line search with weak Wolfe conditions. These algorithms are run on the Rosenbrock function, Logistic Regression on MNIST, and the Booth function, where Newton's function performed the best as it converged in the fewest number of iterations for all the objective functions. However, Newton's method is computationally expensive as the dimension of the problem increases. DFP and BFGS are potential alternatives with the right choices of parameters for the Wolfe conditions, while steepest descent is the least expensive algorithm due to its simplicity, yet may take far more iterations to converge.

2. Methods

Bisection line search with weak Wolfe conditions was used to determine the step size, ensuring sufficient decrease without steps too large (W1) or too small (W2). Using parameters $0 < c_0 < c_1 < 1$, descent direction d^k, and step size α_k, the Wolfe Conditions are defined as

$$f(x^k + \alpha^k d^k) \leq f(x^k) + c_0 \alpha^k (d^k)^T \nabla f(x^k), \tag{W1}$$

$$c_1 (d^k)^T \nabla f(x^k) \leq (d^k)^T \nabla f(x^k + \alpha^k d^k). \tag{W2}$$

The bisection line search method is initialized with interval at $a = 0$, $b = \infty$, and initial step size $t = t_0$. If (W1) is not satisfied, the upper bound of the interval is decreased with $b = t$ and $t = \frac{b+a}{2}$. If (W2) is not satisfied, the lower bound of the interval is increased with $a = t$ and $t = \min\{2a, \frac{b+a}{2}\}$. When both conditions are satisfied, the max iterations is reached, or $|a - b| < 10^{-10}$, then the step size is set as $\alpha_k = t$. When $f(x^k + td^k k)$ is nan, then the current step size may render $f(x^k + td^k k)$ invalid, stopping the algorithm from continuing.

The descent algorithms are defined as $x^{k+1} = x^k + \alpha^k d^k$ where $d^k = -B(x^k)\nabla f(x^k)$, with $s^k := x^{k+1} - x^k$ and $y^k = \nabla f(x^{k+1}) - \nabla f(x^k)$, and the matrix $B(x^k)$ is chosen as

- (Steepest descent or SD) $B(x^k) = I$
- (Newton) $B(x^k) = \nabla^2 f(x^k)^{-1}$
- (DFP) $B(x^{k+1}) = B(x^k) - \frac{\left(B(x^k)y^k\right)\left(B(x^k)y^k\right)^T}{(y^k)^T B(x^k) y^k} + \frac{s^k (s^k)^T}{(y^k)^T s^k}$
- (BFGS) $B(x^{k+1}) = \left(I - \frac{s^k (y^k)^T}{(y^k)^T s^k}\right) B(x^k) \left(I - \frac{y^k (s^k)^T}{(y^k)^T s^k}\right) + \frac{s^k (s^k)^T}{(y^k)^T s^k}$.

Steepest descent defines the direction of the negative gradient to maximize the decrease in each step. Newton's method utilizes $(\nabla^2 f(x^k))^{-1}$ and can achieve quadratic convergence when sufficiently close to a minimizer; however, this renders the method sensitive to initialization and can be computationally expensive. If $\nabla^2 f(x^k)$ is not positive definite, d^k may not work, so a steepest descent direction is used in this report. Another approach is replacing $\nabla^2 f(x^k)$ with a positive

Date: February 18, 2025.

definite matrix approximately close, leading to a family of quasi-Newton algorithms. Two of these algorithms implemented in this report is DFP and BFGS, which use a rank-2 update to the matrix.

The objective functions used in report is the Rosenbrock function (RSN), Logistic Regression (LGT) on MNIST(a database of handwritten digits), and the Booth function [1], which was chosen because it is a plate-shaped function with a global minimum at $f(x^*) = 0$ at $x^* = (1, 3)$ with the potential to explore how algorithms converged with different initial points. (Table 1).

Objective	Initialization
$f(x) = \sum_{k=1}^{n}(x_i - 1)^2 + 100\sum_{k=1}^{n-1}(x_i^2 - x_{i+1})^2$	$n = 2, 5, 10, 50$; $x_0 = (-1, ..., -1)$
$f(x) = \sum_{i=1}^{m}\{\log(1 + \exp(\langle a_i, x\rangle)) - b_i\langle a_i, x\rangle\} + \frac{\lambda}{2}\|x\|^2$	$\lambda = 0.001, 0.01, 0.1$; $x_0 = 0$
$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	$x_0 = (-5, -5)$, $x_0 = (-5, 5)$
Variables: $c_0 = 0.1$, $c_1 = 0.4$, $t_0 = 1$, max_iter = 5000, tol = 1e-5	

Table 1. Objective functions, their initialization and initial parameter values.

3. Results

The four descent algorithms were run on the three objective functions, and the number of iterations and cumulative time to run each algorithm was recorded (Table 2).

Total Iter.	Rosenbrock				Logistic Regression			Booth	
Method	$n = 2$	$n = 5$	$n = 10$	$n = 50$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$x_0 = (-5, -5)$	$x_0 = (-5, 5)$
SD	2232	3498	3980	4446	352	350	269	12	43
Newton's	20	19	39	85	17	15	13	1	1
DFP	47	93	92	187	5000	2631	434	14	51
BFGS	21	61	69	109	435	253	134	12	48

Total Time	Rosenbrock				Logistic Regression			Booth	
Method	$n = 2$	$n = 5$	$n = 10$	$n = 50$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$x_0 = (-5, -5)$	$x_0 = (-5, 5)$
SD	666.5	2025	2114	2034	148.3	157.2	135.9	0.0198	0.1212
Newton's	0.251	0.030	0.395	3.673	10.56	7.828	5.985	0.0	0.0
DFP	1.103	2.086	1.612	8.303	141967	44256	1291	0.139	0.177
BFGS	0.281	0.380	0.720	3.140	2498	844.4	209.8	0.086	0.141

Table 2. (Top) Total number of iterations for each objective with $n = 2, 5, 10, 50$ for RSN, $\lambda = 0.001, 0.01, 0.1$ for LGT, and $x_0 = (-5, -5)$ and $x_0 = (-5, 5)$ for the Booth function, respectively. (Bottom) Total time (s) for each descent method to run for each objective function parameter.

For RSN, SD took the longest amount of time to run and converge, with its total number of iterations on the order of 10^3, while also taking much more time (Table 2). It is followed by DFP, BFGS, and Newton converging in the fewest number of steps. As seen in Figure 1, the gradient norm for Newton rapidly decreased within a few steps as it became close enough to the minimizer. As n increased, i.e. more dimensions were introduced, all the methods took more steps and time to run. Note that for $n = 50$, Newton took more time than BFGS to run, despite converging in fewer steps (Table 2).

For LGT, the number of iterations to converge decreased as λ increased (Table 2). DFP could not run with $c_1 = 0.4$ as the step size became too small, so it was adjusted to $c_1 = 0.25$. This method took the longest to run and did not converge for $\lambda = 0.001$, even when max iterations was increased to 10000 because the gradient norm ended up oscillating between 10^{-2} and 10^{-3}. BFG was also altered with $c_0 = 0.01$ and $c_1 = 0.7$ to produce the plots. Newton's method was again

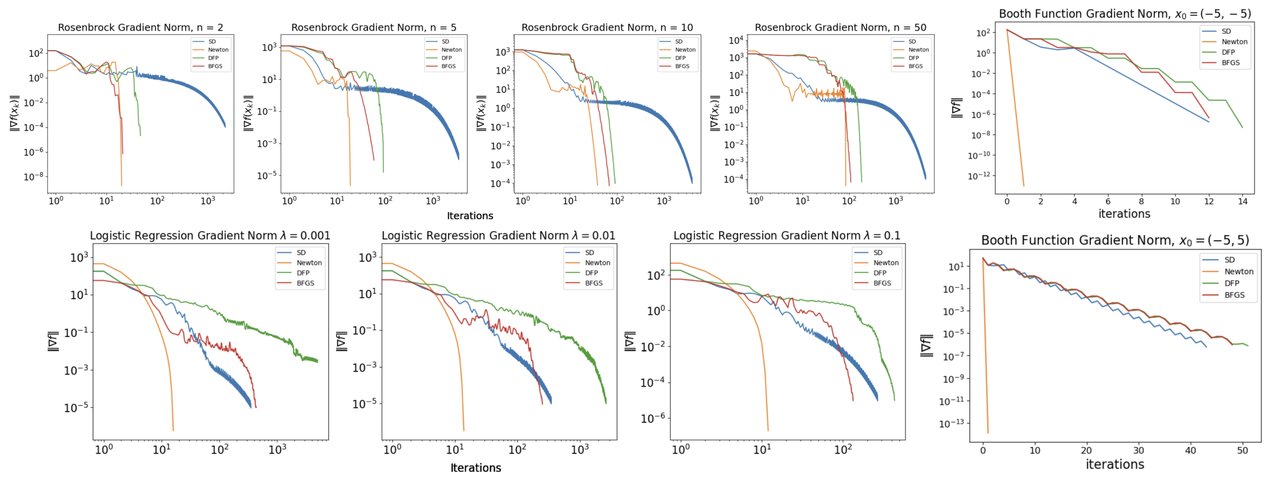

FIGURE 1. Comparison between each descent algorithm for gradient norms of RSN (top row, first four), LGT (bottom row, first three), and Booth (last column).

the fastest to converge step-wise. Between SD and BFGS, SD converged in fewer iterations for $\lambda = 0.001$ while needing more steps for $\lambda = 0.01, 0.1$, yet the cumulative time it took for SD to run was much faster than BFGS for all λ (Table 2).

All four methods ran the fastest with the Booth function, where Newton's methods even converged within a single step for both initial conditions. For the other methods, they converged faster with the first initial point set at $x_0 = (-5, -5)$ compared to $x_0 = (-5, 5)$. The other methods also performed similarly with similar rates of decrease, with DFP taking the most steps overall. For $x_0 = (-5, -5)$, DFP also taking almost double (or more) the amount of time to run compared to the other methods despite only being two steps more (Table 2).

For all three functions, the function value monotonically decreased over time. However, the gradient norm was jagged with either an increase and decrease in each step, even as it lead to a overall decrease over all the iterations. This is likely due to the gradient changing to find an optimal point where the norm may increase, but resulting in a larger decrease in the next step.

4. Discussion and Conclusions

Newton's descent was the best method between all three objective functions as it converged in the least number of iterations with a potential lowest overall running time. It performed best for the Booth function that was a quadratic function of two variables, with a constant $\nabla^2 f(x)$. However, as the complexity or dimensions of the problem increased, Newton's became more computationally expensive where the time it took to ran the method increased.

BFGS was the next best choice for RSN as it converged in fewer steps after Newton's, while also taking less amount of time. However, BFGS, and DFP in particular, was harder to initialize as they were sensitive to the initial values chosen for c_0 and c_1 depending on the objective function. Several parameter values were considered for these algorithms on LGT before finding one that did not run into an error, settling on a smaller c_2 value to ensure the step size did not get too small. These two methods also took more time with LGT and Booth due to their complexity in calculating $B(x^k)$. Steepest descent can be faster to run each step compared to DFP or BFGS due to its simplicity, but it may take more iterations to run overall as the decrease in each step is not always ideal.

A future direction to explore is testing more values of c_0 and c_1 to see if DFP and BFGS can be optimized for LGT to get convergence. It would also be interesting to test these algorithms on other types of objective functions, such as those with many local minima or with steep drops and see how well each one performs.

Acknowledgments

Thank you to Prof. Bamdad Hosseini for the discussions on different descent, line search algorithms, and the Wolfe conditions, and also thank you to the TA Alex Hsu for providing clarification the Wolfe conditions and how to implement them. Furthermore, thank you to peers Christina Nicolaides and David Kieke for the help in coding parts of the descent algorithms.

References

[1] S. Surjanovic and D. Bingham. Booth function. `https://www.sfu.ca/~ssurjano/booth.html`, 2023. Accessed: 2025-02-16.

Appendix

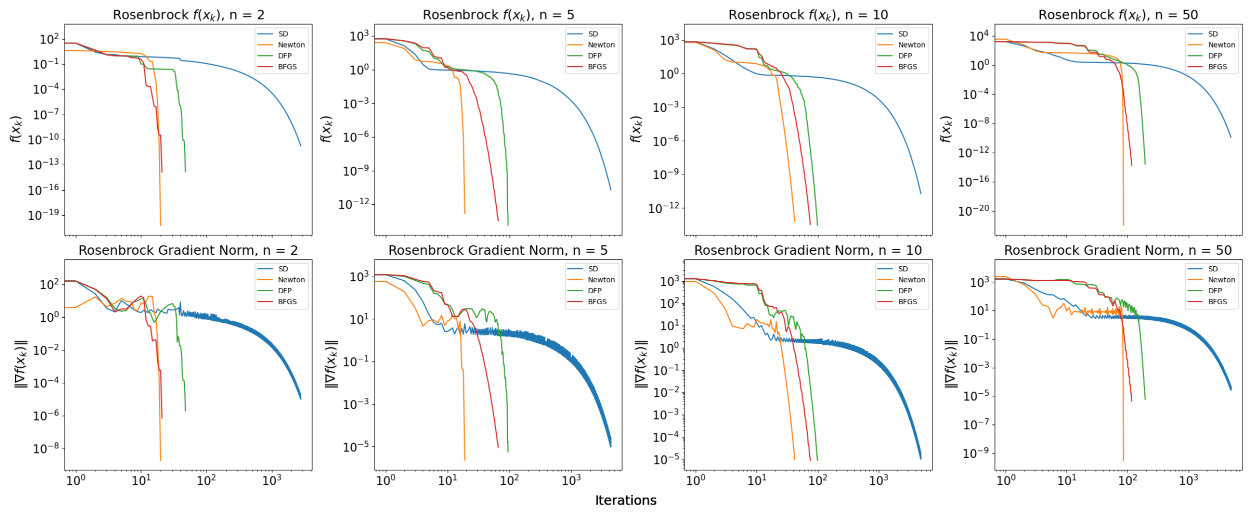

FIGURE 2. Comparison of function values (top row) and gradient norms (bottom row) of the descent methods for each $n = 2, 5, 10, 50$ of the Rosenbrock function.

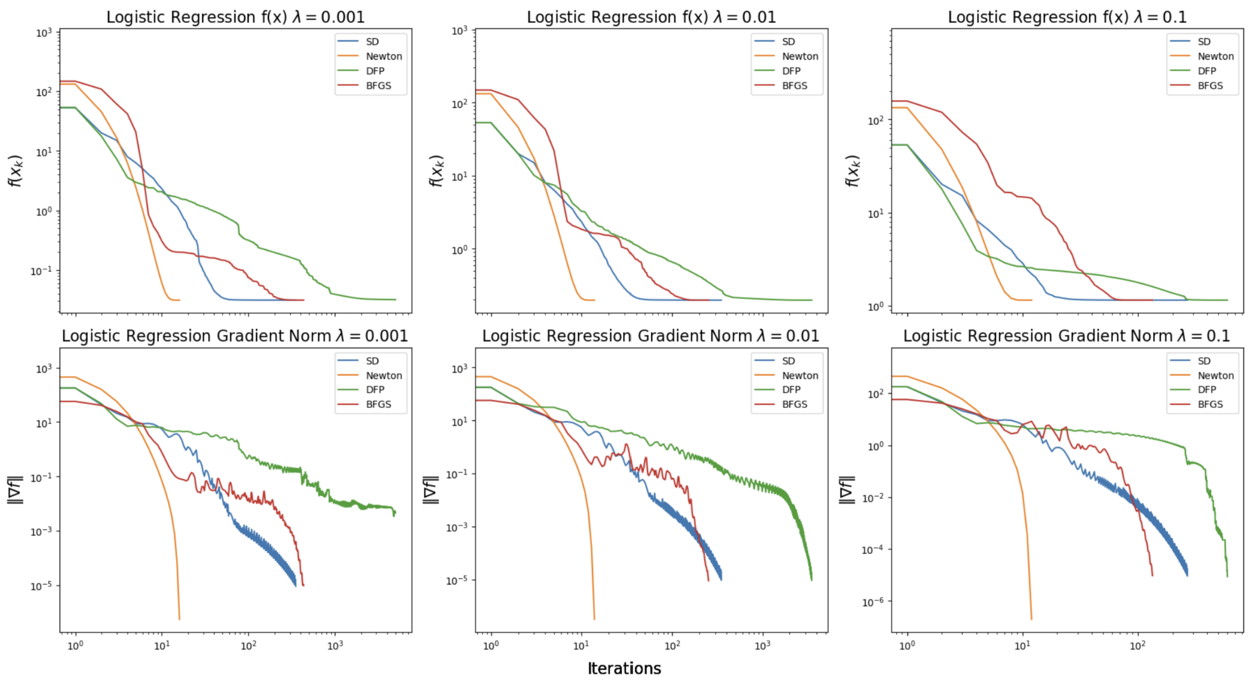

FIGURE 3. Comparison of function values (top row) and gradient norms (bottom row) of the descent methods for each $\lambda = 0.001, 0.01, 0.1$ of the Logistic Regression function.

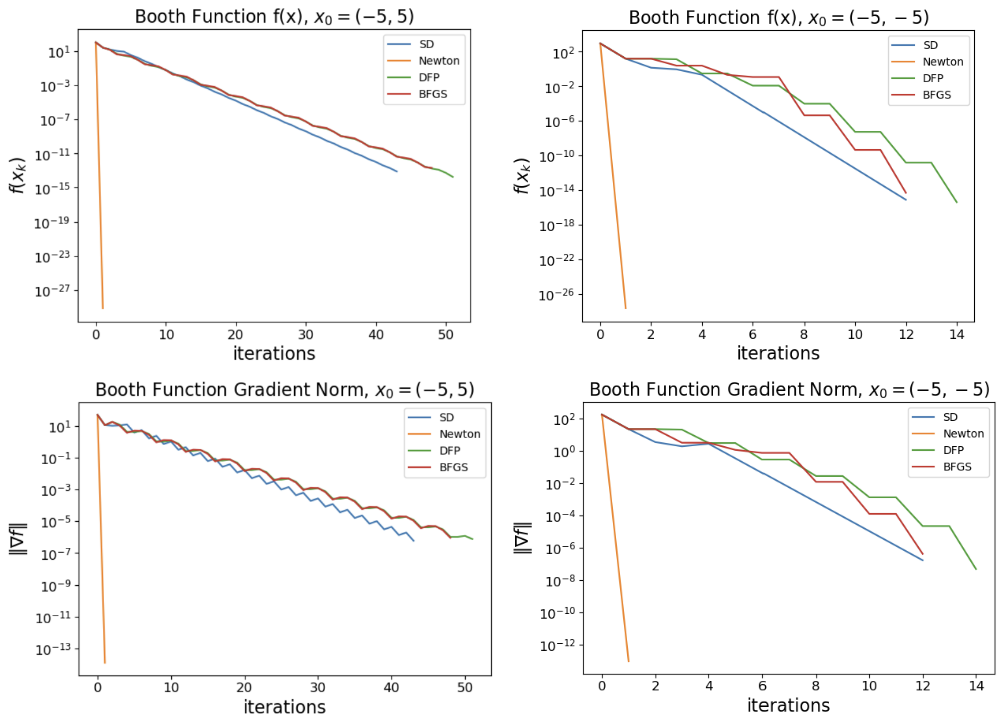

FIGURE 4. Comparison of function values (top row) and gradient norms (bottom row) of the descent methods for each $x_0 = (-5, -5)$ (left) and $x_0 = (-5, 5)$ (right) of the Booth function.